Overfitting Vs Underfitting

**Overfitting:**

Overfitting happens when a machine learning model learns the training data too well, including its noise and random fluctuations, instead of just learning the general patterns. As a result, while the model performs very well on the training data, it struggles to perform well on new, unseen data.

**How to Identify:**

* Low accuracy on both training and testing data.
* The model fails to capture patterns and trends in the data, resulting in poor performance across the board.

**Avoiding Overfitting**

* Simplify the Model:

Use a model that is less complex. Lesser the attributes the better

* Use Regularization:

Apply techniques that add a penalty to the model if it becomes too complex. This helps in keeping the model simple.

* Cross-Validation:

Test your model on different subsets of data to check if it performs well in various situations, not just on the data it was trained on.

* Early Stopping:

What It Means: Stop training your model when it starts to perform worse on a separate validation set, even if it’s still improving on the training data.

**Underfitting**

Underfitting occurs when a machine learning model is too simple to capture the underlying patterns in the data. As a result, the model performs poorly on both the training data and new data because it fails to learn the important relationships.

**How to Identify:**

* Low accuracy on both training and testing data.
* The model fails to capture patterns and trends in the data, resulting in poor performance across the board.

**Avoiding Underfitting**

* Increase Model Complexity:

Use a more advanced model or add more features to capture the details in the data.

* Feature Engineering:

Create or choose better features that help the model understand the data more effectively.

* Train Longer:

Allow more time for the model to learn from the data.

**What is Accuracy, Precision, Recall and F1 score and when are these used**

**Accuracy**

* **What It Tells You:** How often your model is correct.
* **When to Use It:** When you care about overall correctness and your data is balanced

**Precision**

* **What It Tells You:** When your model predicts something as positive (like predicting "spam" in emails), how often is it actually correct?
* When to Use It: When it's more important to avoid false alarms (false positives).

**Recall**

* **What It Tells You:** Out of all the actual positive cases (like all the real spam emails), how many did your model correctly identify?
* **When to Use It:** When missing a positive case is costly. For example, in medical testing, you want to catch as many positive cases as possible, even if it means getting some false positives.

**F1 Score**

**What It Tells You:** It's a balance between precision and recall. It's useful when you need a balance between catching all positive cases (recall) and ensuring that your positive predictions are accurate (precision).

**When to Use It:** When you have an imbalanced dataset, and both precision and recall are important to you.